CS 3360: Programming Language Concepts
Spring 2025

CRN: 23423
Lecture: TR 3:00 PM - 4:20 PM in BUSN 309
Instructor: Yoonsik Cheon (ycheon@utep.edu); office hours: TR 1:30 PM - 2:30 PM in CCSB 3.0606
Teaching assistant: TBA
Prerequisite: CS 2302 with a grade of C or better (Recommended: CS 3331 and CS 3432)

Course Objectives

In this course, we will explore the world of programming languages, focusing on both theoretical foundations and practical applications. Our primary goal is to equip you with the tools needed to critically evaluate and quickly master various programming languages and constructs.

- We will maintain a balance between theoretical understanding and hands-on practice. You will survey various constructs and capabilities found in modern programming languages, examining design trade-offs and implementation considerations.
- By understanding the broad spectrum of language features and design alternatives, you will be well-prepared to adapt to new programming languages throughout your professional career. This knowledge is essential for navigating the challenges of learning and using new languages.
- The course will lay a solid foundation for advanced studies in areas like compilers and programming language semantics, providing a deeper insight into the implications of design choices.
- To illustrate general programming language characteristics, we will explore several languages in detail:
    - Dart: A modern object-oriented language.
    - Haskell: A functional programming language.
    - PHP: A web scripting language.
    - Prolog: A logic programming language.
- You will undertake a programming project in each of these languages, giving you practical experience with different programming paradigms and reinforcing your learning through hands-on practice.

By the end of this course, you will have a deep understanding of various programming languages, the ability to evaluate and learn new languages efficiently, and practical experience with multiple programming paradigms.

Textbooks

Our required textbook for this course is *Concepts of Programming Languages* by Robert W. Sebesta, 12th edition, published by Addison Wesley in 2018. This book is available at the UTEP bookstore and is crucial for the course, as reading assignments will be assigned from this book. Additionally, we suggest the following supplementary books:

- Kevin Tatroe and Peter MacIntyre. *Programming PHP: Creating Dynamic Web Pages*. Fourth edition, O'Reilly, 2020. Focus on Chapters 1-6.
- Gilad Bracha. *The Dart Programming Language*. Addison-Wesley Professional, 2016.
- Will Kurt. *Get Programming with Haskell*. Manning Publications, 2018. Specifically, Units 1, 2, and 4.

Electronic versions of these recommended books are available for free through the UTEP Library. If you are accessing the library from outside the UTEP domain, you may need to use a VPN. These textbooks and resources will support your learning by offering a blend of theoretical knowledge and practical applications, aligned with the course objectives.

Examinations

This course includes both a mid-term exam and a final exam. The mid-term exam is scheduled during a regular class session and will last for 80 minutes, while the final exam aligns with the university's specified date. A makeup exam will be considered only in exceptional or unavoidable situations, such as incapacitating illness or attendance at a conference for presentation purposes. Should you encounter such circumstances requiring a makeup exam, it is

imperative to promptly inform the course staff. For those planning to attend a conference for a paper presentation, arrangements for the exam must be made well in advance. Regardless of the situation, eligibility for a makeup exam is contingent upon providing an official document that explains your circumstances.

Assignments

This course comprises three distinct assignment types: reading, homework, and programming. It is advisable to allocate approximately 3-4 hours per week for reading and homework assignments, along with an average of 3 hours per week dedicated to programming tasks. Each programming assignment is anticipated to require 8-10 hours of effort, potentially resulting in a more demanding workload during weeks when programming assignments are due.

- Reading assignments: These assignments involve delving into the textbook to prepare for upcoming lessons. Following these readings, quizzes will be administered to ensure the completion of weekly reading tasks and comprehension of core concepts from recent lessons. Typically, quizzes will take around 10 minutes each and will cover material assigned for upcoming lessons as well as selected content from prior sessions. It is essential to note that there will be no opportunities for makeup quizzes, as answers will be accessible after the designated due dates.
- Homework assignments: These assignments entail completing exercises from the textbook's chapters. These assignments are designed to reinforce the material covered in lectures and provide additional practice. Assignments involving material not covered in lectures will be generously evaluated to encourage independent learning and exploration of the subject matter.
- Programming assignments: These assignments provide hands-on experience in specific languages and programming paradigms. Throughout the course, you will work with PHP for web scripting, Dart for contemporary object-oriented programming, and Haskell for functional programming. These assignments are crucial for developing practical skills and understanding the real-world application of programming concepts.

It is mandatory to complete all assignments individually unless explicitly stated otherwise. While engaging in general discussions with peers is encouraged, the actual composition and writing of text or code must be done independently. Additionally, avoid copying and pasting text or code from the Internet or generative AI; instead, strive to rephrase content in your own words. If you need assistance, do not hesitate to approach the course staff. All work must be submitted through Blackboard, and late submissions will only be considered if arrangements are made in advance or if extraordinary circumstances warrant an exception.

Grading Policy

Your grade is individual and is not influenced by the grades of your peers. Our grading system does not involve curving; every student has the potential to achieve an A. The grading process aims to maintain a standard of excellence and provide constructive feedback, rather than comparing students to each other. Your final letter grade will be determined through a combination of factors, including lessons (readings and exercises), homework assignments, programming tasks, and exams. The approximate percentage breakdowns are as follows:

Activities	Percent (%)
Lessons (readings and exercises)	25
Homework assignments	25
Programming assignments	25
Exams	25

Additionally, up to 5% bonus points are available for class attendance and active participation. To earn these extra points, you must attend lectures punctually and engage in class discussions constructively and well-prepared. This includes asking or answering questions that reflect your reading efforts and attempting to comprehend the material. Meeting deadlines for classwork and activities is equally important. Participation will be assessed through Blackboard tracking tools, discussions, and group assignments.

Adherence to due dates is vital. There will be no provision for makeup assignments or late submissions unless a substantial and compelling reason, sanctioned by the course staff, exists. Unless otherwise directed, all coursework should be submitted electronically through Blackboard. Ensure that your work is submitted by the specified due date

or secure special permission from the course staff prior to the deadline. Extensions beyond the subsequent assignment will not be granted, except under exceptional circumstances.

Final letter grades will be determined by computing the percentage of total points you have accumulated. The nominal conversion from percentage score to letter grade is as follows:

Letter grade	Percent (%)	Performance	
A	90-100	Excellent	
B	80-89	Good	
C	70-79	Average	
D	60-69	Poor	
F	0-59	Failing	

The instructor reserves the right to adjust these criteria downward by considering overall class performance, such as designating an A for scores of 88% or higher.

Attendance/Participation Policy

Consistent and punctual class attendance is crucial to your success in this course. Regular attendance is not only expected but mandatory, as it significantly impacts your academic progress. The instructor reserves the right to impose penalties for unexcused absences, including reducing your final grade by one point for each unexcused absence beyond three. The following excerpt is from the 2024-2025 Catalog:

"The student is expected to attend all classes and laboratory sessions, and attendance is mandatory for all freshman-level courses. It is the responsibility of the student to inform each instructor of extended absences. When, in the judgment of the instructor, a student has been absent to such a degree as to impair his or her status relative to credit for the course, the instructor can drop the student from the class with a grade of W before the course drop deadline and with a grade of F after the course drop deadline."

Adhering to regular attendance aligns with academic expectations and ensures compliance with university policies. This commitment will significantly enhance your learning experience and overall performance on the course. Make sure to inform the course staff of any extended absences to avoid any penalties and to maintain your academic standing.

Standards of Conduct

Maintaining a professional and respectful demeanor is essential to fostering a positive learning environment. All graded assignments, including reading, exercises, homework, and exams, must be completed independently and reflect your own work. While discussing general ideas with peers is encouraged, presenting material copied from external sources—such as individuals, books, websites, or generative AI—is strictly prohibited.

Plagiarism, as defined by university policy, is the unauthorized use or reproduction of someone else's work or ideas without proper acknowledgment. This includes submitting work that was not created by you, reusing work from other courses without permission, or presenting borrowed content as your own. Instructors are required to report any suspected instances of plagiarism or academic dishonesty to the Office of Community Standards (formerly the Office of Conduct and Conflict Resolution) for investigation and appropriate action. Plagiarism is a serious violation of academic integrity and will not be tolerated.

It is an institutional requirement for instructors to report any suspected instances of plagiarism and academic dishonesty to the Office of Community Standards (formerly the Office of Conduct and Conflict Resolution) for investigation and appropriate action. Plagiarism is a serious violation of academic integrity and will not be tolerated.

The use of generative AIs and large language models (LLMs) is encouraged as they serve as excellent tools for learning, particularly in programming and mastering programming languages. These tools can help with conceptual understanding, exploring alternative approaches, and generating sample code. However, all submitted work, including code, must be created by you, the student. While LLMs can offer valuable guidance and clarification, your final submissions must demonstrate your own understanding, effort, and mastery of the material.

Upholding the highest standards of academic integrity is critical to maintaining a fair and honest educational environment, and all students are expected to adhere to these principles.

Accommodations

If you have a disability and require classroom accommodations, please contact the Center for Accommodations and Support Services (CASS) as soon as possible. It is important that you reach out to CASS promptly to ensure that appropriate accommodations are arranged in a timely manner. You can contact CASS by phone at 747-5148 or via email at cass@utep.edu. Their office is at UTEP Union East, Room 106.

For more information and to access additional resources, please visit the CASS website: http://www.sa.utep.edu/cass.

Please note that the instructor is only able to provide accommodations listed in the accommodation letter provided by CASS. Any additional accommodations not specified in the letter cannot be granted. It is your responsibility to ensure that the instructor receives a copy of the accommodation letter at the beginning of the course and to discuss it with the instructor to facilitate the timely implementation of necessary accommodations.

Schedule

The table below presents a planned schedule for the course. For the most current schedule, consult the course website.

Dates		Topics	Readings	Assignments
Week 1	Jan. 21, 23	About CS 3360		
Week 2	Jan. 28, 30	Preliminaries Describing syntax	Chapter 1 Sections 3.1-3.3	Homework 1
Week 3	Feb. 4, 6	Describing syntax Attribute grammar	Section 3.4	
Week 4	Feb. 11, 13	Web scripting with PHP	E-book	Programming 1
Week 5	Feb. 18, 20	PHP: Basic and advanced		
Week 6	Feb. 25, 27	PHP: Modern Names, bindings, and scopes	Chapter 5	Homework 2
Week 7	Mar. 4, 6	Names, bindings, and scopes Data types	Sections 6.1-6.9	Homework 3
	Mar. 11, 13	Spring break		
Week 8	Mar. 18, 20	**Exam 1** Object-oriented programming	Sections 12.1-12.6	
Week 9	Mar. 25, 27	Dart: Part 1	E-book	
Week 10	Apr. 1, 3	Dart: Parts 2 and 3		Programming 2
Week 11	Apr. 8, 10	Dart: Part 3 Functional programming	Sections 15.1-15.3	
Week 12	Apr. 15, 17	Haskell: Part 1	Section 15.8 E-book	
Week 13	Apr. 22, 24	Haskell: Part 2		Programming 3
Week 14	Apr. 29, May 1	Describing semantics Subprograms	Section 3.5 Sections 9.1-9.6	
Week 15	May 6, 8	Logic programming and Prolog	Chapter 16	
Week 16	May 15	**Final** at 4:00 pm – 6:45 pm		

Important Dates

January 20: Dr. Martin Luther King, Jr. holiday – University closed
January 21: Classes begin
February 5: Census day
March 10-14: Spring break
March 18: Exam 1
March 28: Cesar Chavez holiday – No classes
April 4: Drop/withdrawal deadline
April 18: Study day
May 8: Last day of classes
May 9: Dead day
May 15: Final on Thursday at 4:00 pm – 6:45 pm

CS 3360: Programming Language Concepts
Learning Outcomes

Level 1: Knowledge and Comprehension

Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. The material has been presented only at a superficial level. Upon successful completion of this course, students will be able to:

1a. Describe broad trends in the history of the development of programming languages.
1b. Explain the stages of programming language interpretation and compilation.
1c. Understand data and control abstractions of programming languages.
1d. Understand how the attribute grammars describe static semantics.
1e. Describe ways to formally specify the dynamic semantics of small subsets of programming languages, such as expressions and control structures.
1f. Understand code snippets written in a paradigm beyond imperative, object-oriented, and functional, e.g., algebraic, aspect-oriented, logic, or probabilistic languages.
1g. Explain language constructs that promote information hiding and ensure representation independence.

Level 2: Application and Analysis

Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able to:

2a. Define the syntax of a small subset of a programming language using BNF.
2b. Compare different approaches to naming, storage bindings, typing, scope, and data types.
2c. Analyze design dimensions of subprograms, including parameter passing methods, sub-programs as parameters, and overloaded subprograms.
2d. Be able to write programs to solve simple problems in a purely functional language.
2e. Be able to write programs to solve simple problems in a scripting language.

Level 3: Synthesis and Evaluation

Level 3 outcomes are those in which the students can apply the material in new situations. This is the highest level of mastery. Upon successful completion of this course, students will be able to:

3a. Critically evaluate type system options such as static, dynamic, gradual, and optional and features including type inference, polymorphism, and subtyping of programming languages.
3b. Analyze class inheritance and related code reuse mechanisms in object-oriented programming languages, considering choices such as single vs. multiple inheritance, interface vs. implementation inheritance, and static vs. dynamic dispatch, and mixins.
3c. Assess the utility of advanced expression syntax, including lambda expressions and higher-order functions, in modern programming languages
3d. Analyze advanced language constructs such as pattern matching, closures, continuations, and concurrency constructs, and critically evaluate their design trade-offs in comparison to traditional programming constructs.
3e. Choose a suitable programming paradigm and language for a given problem or domain.